**Overview**

These scripts are designed to facilitate training and inference tasks using the MaskFormer model, a state-of-the-art approach to instance segmentation. The scripts support custom configurations via YAML files, making it easier to adapt the process to different datasets and environments.

**File Structure**

1. **MaskFormer\_Training.py**:
   * Handles the training of the MaskFormer model on a custom dataset.
   * Includes dataset preparation, model configuration, training, and validation loops.
   * Outputs trained models and relevant metrics.
2. **MaskFormer\_Inference.py**:
   * Uses the trained MaskFormer model to run inference on a test dataset.
   * Calculates performance metrics such as accuracy and IoU (Intersection over Union).
   * Visualizes the results, including confusion matrices and confidence matrices.

**Configuration Files**

The scripts rely on YAML configuration files to specify parameters for training and inference. Here’s an overview of the key settings:

* **user\_config\_train.yaml**: Configuration for training the model.
  + **DATADIR**: Root directory of the dataset.
  + **EPOCHS**: Number of training epochs.
  + **IMGDIR, SEGDIR**: Directories for images and segmentation labels.
  + **IMAGE\_SIZE\_X, IMAGE\_SIZE\_Y**: Image dimensions.
  + **LEARNING\_RATE**: Learning rate for the optimizer.
  + **BATCH\_SIZE**: Batch size for training.
  + **ID2LABEL, LABEL2ID**: Mappings between class IDs and labels.
* **user\_config\_test.yaml**: Configuration for inference.
  + **DATADIR**: Root directory of the test dataset.
  + **TEST\_IMGDIR, TEST\_SEGDIR**: Directories for test images and segmentation labels.
  + **MODEL\_LOCATION**: Path to the trained model to be used for inference.

**Training Workflow (MaskFormer\_Training.py)**

1. **Dataset Preparation**:
   * The dataset is loaded using the TransformerDataSet class, which handles image and segmentation label loading.
   * The dataset is split into training and validation subsets.
2. **Model Configuration**:
   * The MaskFormer model is configured based on the settings in the configuration file.
   * Data augmentation techniques are applied using the albumentations library.
3. **Training Loop**:
   * The model is trained over multiple epochs, with the loss calculated and optimized at each step.
   * Validation is performed after each epoch to monitor performance.
4. **Saving Outputs**:
   * The trained model and processor are saved for later use.
   * Loss plots are generated and saved.

**Inference Workflow (MaskFormer\_Inference.py)**

1. **Loading the Model**:
   * The trained MaskFormer model and processor are loaded from the specified location.
2. **Dataset Preparation**:
   * The test dataset is loaded using the TransformerDataSet class.
3. **Inference**:
   * Inference is performed on the test dataset, generating predicted segmentation masks.
   * Metrics such as overall accuracy, mean accuracy, and mean IoU are calculated.
4. **Visualization**:
   * Confusion matrices and confidence matrices are generated and saved.
   * Sample predictions are visualized alongside the ground truth annotations.

**Key Functions**

* **Training Script (MaskFormer\_Training.py)**:
  + TransformerDataSet: Custom dataset class for loading and handling image-segmentation pairs.
  + train\_val\_dataset: Splits the dataset into training and validation sets.
  + create\_loss\_plots: Generates plots for training and validation loss.
* **Inference Script (MaskFormer\_Inference.py)**:
  + load\_config: Loads configurations from a YAML file.
  + calculate\_overall\_accuracy, calculate\_mean\_accuracy, calculate\_mean\_iou: Metrics calculations.
  + compute\_confidence\_matrix: Generates a confidence matrix based on predictions.
  + get\_inference: Runs inference on a single sample from the dataset.

**How to Run**

1. **Training**:
   * Ensure that your dataset is organized and the user\_config\_train.yaml file is properly set up.
   * Run python MaskFormer\_Training.py to start training.
2. **Inference**:
   * Adjust the user\_config\_test.yaml file to point to your test dataset and trained model.
   * Run python MaskFormer\_Inference.py to perform inference and generate results.